1. 题 (8分)

1.1 X 没有看到是 sorted

1.2 X 没有使用 sorted array 的 push.

```
void push(const T &item) {
  assert(size < 10);
  int i = size;
  for (; i >0 && data[i] < item; --size){
    data[i] = data[i-1];
  }
  data[i] = item;
}
```

X data[i-1]

```
// REQUIRES: the MinQueue is not full
// EFFECTS: Adds 'item' to the MinQueue
void push(const T &item) {
  assert(size < 10);
  int i = size;
  while (i > 0 && data[i - 1] < item) {
    data[i] = data[i - 1];
    --i;
  }
  data[i] = item; // Insert to the correct position
  ++size;
}
```

1.3 X 没有理解你是 non-member function

```
template <typename T>
typename MinQueue & MinQueue<T>::operator+=(const T &value){
  this->push(value);
  return *this;
}
```

是 member function.

```
template <typename T>
MinQueue<T> & operator+=(MinQueue<T> &q, const T &t) {
  q.push(t);
  return q;
}
```

2题

2.1 ✓

2.2 ✓

2.3 ✓

3题 12:45 - 12:55 (10分)

3.1 ✓

3.2 X

```
int size = sculptures.size();
for (int i = 0; i < size; ++i){
  delete sculptures[i];
}
```

```
~Gallery() {
  for(size_t i = 0; i < sculptures.size(); ++i) {
    delete sculptures[i];
  }
}
```

正确 vector 前 Loop 都要用 size_t.

3.3 ✓

4题 1:00 - 1:11 (11分)

4.1 ✓

```
int count_matches(const List &other) const {
  int number = 0;
  Node *n1_ptr = this->fisrt;
  Node *n2_ptr = other->first;
  while (n1_ptr&&n2_ptr){
    if (n1_ptr->datum == n2_ptr->datum){
      ++number;
    }
    n1_ptr = n1_ptr->next;
    n2_ptr = n2_ptr->next;
  }
  return number;
}
```

other.first.

4.2

```
template <typename T>
void move_range_to_back(Node *rng_first, Node *rng_last) {

  if (rng_last == last) {
    return; // do nothing if already at back
  }

  rng_last->next->prev = rng_first->prev->next ;

  if (rng_first->prev) {
    rug_fist->prev->next = rng_last->next;
  }
  else {
    first = rng_last->next ;
  }

  last->next = rng_first ;

  rng_first->prev = last;

  rng_last->next = nullptr ;

  last = rng_last;

}
```

rng_first->prev

5题 2:44 - 2:58 (14)

5.1 ✓

Complete the implementation of `Replacer` below by filling in the boxes.

If you believe a blank/box should be **empty**, write `BLANK`.

```
template <typename T>; // Note: T must support ==
class Replacer {
private:
  const T target;
  const T replacement;

public:
  Replacer( T target, T replacement
    : target(target), replacement(replacement) { }

  // Function call operator
  T operator()( const T &
    item) const {
    if (item == target){
      return replacement;
    }
    else {
  }
};
```

const T& target. const T& replacement.

5.2 ✓

```
for (InIter it = begin; it != end; ++it){
    *out = fn(*it);
    ++out;
}
```

```
while(begin != end) {
  *out = fn(*begin);
  ++begin;
  ++out;
 }
```

5.3 X 废题.

```
void replaceAll(vector<string> &words, const string &target, const string &replacement) {
vector<string> v_tempt = words;
map(words.begin(),words.end(),Replacer(target,replacement),v_tempt.begin());
words = v_tempt;
```

```
void replaceAll(vector<string> &words, const string &target, const string &replacement) {
  Replacer rep(target, replacement);
  map(words.begin(), words.end(), rep, words.begin());
}
```

6题 3:12 - 3:17

6.1 ✓

```
if (node->datum == value){
  return index;
}
else if (!node){
  return -1;
}
else {
  return  index_of_helper(node->next, value, ++index);
}
```

7题 3:19 - 3:21

7.1 ✓

7.2 ✓

8题 3:25 - 3:55

8.1 X

```
template<typename T>
int DoubleTree<T>::Node::count(T datum) const {
  if (datumCount == 0) { return 0; }
  if (datumCount == 1) {
    return datum1 == datum ? 1 : 0;
  }
  if (datum < datum1) {
    return left->count(datum);
  }
  int count = 0;
  if (datum2 <= datum) {
    count += right->count(data);
  }
  if (datum1 == datum) { ++count; }
  if (datum2 == datum) { ++count; }
  return count;
}
```

8.2 X

```
int DoubleTree<T>::count(T datum) const
  return root->count(datum);
```

考虑 root 指向 nullptr.

```
int DoubleTree<T>::count(T datum) const
  if (!root) {
    return 0;
  }
  return root->count(datum);
}
```